Neural Processing Letters
https://doi.org/10.1007/s11063-021-10424-x

Illustrative Discussion of MC-Dropout in General Dataset: Uncertainty Estimation in Bitcoin

Ismail Alarab[1] · **Simant Prakoonwit**[1] · **Mohamed Ikbal Nacer**[1]

Accepted: 1 January 2021

Abstract

The past few years have witnessed the resurgence of uncertainty estimation generally in neural networks. Providing uncertainty quantification besides the predictive probability is desirable to reflect the degree of belief in the model's decision about a given input. Recently, Monte-Carlo dropout (MC-dropout) method has been introduced as a probabilistic approach based Bayesian approximation which is computationally efficient than Bayesian neural networks. MC-dropout has revealed promising results on image datasets regarding uncertainty quantification. However, this method has been subjected to criticism regarding the behaviour of MC-dropout and what type of uncertainty it actually captures. For this purpose, we aim to discuss the behaviour of MC-dropout on classification tasks using synthetic and real data. We empirically explain different cases of MC-dropout that reflects the relative merits of this method. Our main finding is that MC-dropout captures datapoints lying on the decision boundary between the opposed classes using synthetic data. On the other hand, we apply MC-dropout method on dataset derived from Bitcoin known as Elliptic data to highlight the outperformance of model with MC-dropout over standard model. A conclusion and possible future directions are proposed.

Keywords Uncertainty estimation · MC-dropout · Bayesian approximation · Risk and uncertainty · Bitcoin data

Ismail Alarab: Please note that the LNCS Editorial assumes that all authors have used the western naming convention, with given names preceding surnames. This determines the structure of the names in the running heads and the author index.

✉ Ismail Alarab
ialarab@bournemouth.ac.uk

Simant Prakoonwit
sprakoonwit@bournemouth.ac.uk

Mohamed Ikbal Nacer
mnacer@bournemouth.ac.uk

[1] Bournemouth University, Poole, UK

Published online: 16 January 2021

1 Introduction

Deep neural networks (DNNs) have attained successful performance in variety of fields such as medical imaging [1], computer vision [2], fault detection [3], and attractively in big data applications [4]. DNNs are a powerful tool to learn parameters to find the best point estimation that maps the given feature vector to the desired output in classification tasks. Due to the point estimation, these mappings are assumed to be exact which is not always the case in the presence of softmax function that outputs single predictions. Consequently, the misclassified examples are often provided erroneously with unjustified overconfidence. On the other hand, Bayesian approaches such as Bayesian neural networks (BNNs) and Gaussian processes have received significant attention to provide uncertainty measurements besides the predictive probabilities. Unlike single predictions, BNNs and Gaussian processes yield predictive distributions in which the weights of BNNs are incorporated with priors distribution [5], whereas Gaussian processes introduces priors over functions.

In other words, Gaussian processes sample prior functions from a multivariate Gaussian distribution and update these priors with a newly collected data via Bayesian rule. BNNs and Gaussian processes are computationally expensive when dealing with a high number of parameters as the case of neural networks with a high number of units and multiple layers. BNNs require to get the posterior distribution across the network's parameters, in which all possible events are obtained at the output. Gaussian processes need to sample prior functions from multivariate Gaussian distribution, wherein the dimension of Gaussian distribution increases proportionally with the number of training points involving the whole dataset during predictions. Recently, the outgrowth of Bayesian approaches has witnessed a resurgence of interest which is computationally much efficient than BNNs known as Monte-Carlo dropout (MC-dropout) method. As introduced in [6], MC-dropout is shown to be as an approximation of the probabilistic Bayesian model—deep Gaussian process. Furthermore, this technique is viewed as the minimisation of Kullback–Leibler divergence between an approximate distribution and the posterior of a deep Gaussian process. MC-dropout has achieved promising results in regression and classification problems especially in image datasets [7–9]. Concisely, MC-dropout is a method of performing multiple stochastic forward passes with the means of activated dropout in a neural network during the testing process to provide ensemble of predictions that could reflect uncertainty estimations. Establishing uncertainty is necessary for the model to know what it doesn't know. The major types of uncertainty are known as epistemic, aleatoric, and predictive uncertainty which is the combination of the former types [10]. Epistemic uncertainty is derived from the lack of training data in the region of prediction. This uncertainty is reducible by collecting more data which enhances the model's belief. Aleatoric uncertainty is accounted by noisy observations such as data collected by sensors. This uncertainty is irreducible by the model but rather mitigated by the source of observations.

Apart from uncertainty, another term is introduced in [11] is called risk. It has been defined as the inherent stochasticity in the model's parameters and can be seen as the variability of the decision boundary in classification applications. MC-dropout has been criticised by many researchers regarding the type of uncertainty that is captured especially in [11]. For instance, the work in [11] has claimed that MC-dropout is tied with risk and not uncertainty estimations which contradicts MC-dropout's behaviour in [10]. For this purpose, we aim to provide illustrative experimental discussion about the behaviour of MC-dropout using a toy example in the light of the previous contradictory contributions in [6, 11]. Also, we discuss the performance of MC-dropout on real data derived from Bitcoin

blockchain and we provide the strength and drawbacks of this method. Furthermore, we show that MC-dropout is not able to deal with out-of-distribution data, whereas it works well on some cases which we discuss in the upcoming sections.

2 Overview of the Related Work

The growing interest of BNN models has been extensively established in [12, 13] in which Gaussian distribution is identified over the network's parameters around the mode to compute the posterior distributions. Although Bayesian models are powerful in uncertainty estimation, the exact inference of posterior distribution is intractable. Meanwhile, variational inference, Bayesian training by the Hybrid Monte Carlo method, and Markov Chain Monte Carlo with Hamiltonian Dynamics also exist referring to [14–16] in which assumptions to be made regarding the approximated posterior distribution. These methods are difficult to scale to large datasets [10]. The mentioned studies are limited in some cases or require additional cost. Recently, an efficient method to compute uncertainties is introduced known as Monte-Carlo dropout (MC-dropout) [6]. Regarding classification tasks, we explain the practical implementation of MC-dropout, induced uncertainties, and the criticism of this work in more details in the following sections.

2.1 How Does MC-Dropout Work?

Primarily, dropout is introduced as a simple regularisation technique to reduce overfitting in neural network [17]. This technique is often applied during the training process of neural network in which a less over-fitted and more generalisation of the model over the predictions occur. On the other hand, applying dropout before every weight layer has shown to be mathematically equivalent to an approximation to the probabilistic deep Gaussian process [6].

Consider $\hat{y}$ as an output of neural network model with arbitrary layers L and parameters $w = \{W_1, \ldots, W_L\}$ as the weight matrices. Let be y^* the observed output associated by the input vector x^*. Given a dataset $X = \{x_1, \ldots, x_N\}, Y = \{y_1, \ldots, y_N\}$, the predictive distribution can be expressed as:

$$p(y^*|x^*, X, Y) = \int p(y^*|x^*, w)p(w|X, Y)dw \tag{1}$$

where $p(y^*|x^*, w)$ is the model's likelihood and $p(w|X, Y)$ is the posterior over the weights.

The predictive distribution involves a predictive mean and variance yielding uncertainty estimation. However, the posterior distribution is analytically intractable. Instead, an approximation of variational distribution $q(w)$ is obtained from Gaussian process to be close as much as possible to $p(w|X, Y)$ in which the optimsation process occurs by minimising the Kullback–Leibler divergence (KL) between the preceded distributions as following:

$$KL(q(w)|p(w|X, Y)) \tag{2}$$

With variational inference, the predictive distribution can be approximated as:

$$q(y^*|x^*) = \int p(y^*|x^*, w)q(w)dw \tag{3}$$

Referring to [6], $q(w)$ is chosen to be the distribution over matrices whose columns are randomly set to zero according to Bernoulli distribution expressed as:

$$W_i = M_i.diag\left(\left[z_{i,j}\right]_{j=1}^{K_i}\right) \tag{4}$$

where $z_{i,j} \sim Bernoulli\left(p_i\right)$ for $i = 1, \ldots, L$ and $j = 1, \ldots K_{i-1}$, with K_ixK_{i-1} the dimension of matrix W_i.

p_i refers to the probability of dropout and M_i is a matrix of variational parameters (Please refer to [6] for more details). Thus, drawing T sets of vectors of samples from Bernoulli distribution yields $\left\{W_1^t, \ldots, W_L^t\right\}_{t=1}^{T}$. Therefore, the predictive mean can be written as:

$$E_{q(y^*|x^*)(y^*)} \approx \frac{1}{T}\sum_{t=1}^{T} \hat{y}^*\left(x^*, W_1^t, \ldots, W_L^t\right) = p_{MC}(y^*|x^*) \tag{5}$$

with $\hat{y}^*$ is the mapping of x^* by the given neural network and p_{MC} is predictive mean of MC-dropout, which is equivalent of performing T stochastic forward passes over the neural network during the testing process with dropout then averaging the results. This method is viewed as ensemble of approximated functions with shared parameters which is an approximation of the probabilistic Bayesian approach known as deep Gaussian processes. Since this method yields multiple outputs per input, therefore uncertainty could be inspected by computing the variance, entropy, mutual information…etc. of these outputs; we consider mutual information only in this paper which is discussed later. More practically, the neural network with dropout learns a variety of hypotheses by randomly subsampling from the units of the hidden layer. De-activating dropout during the testing process, the neural network reproduces all hypotheses simultaneously as ensemble of decision functions to perform a single prediction as the case in random forests to reduce the variance and consequently prevent overfitting. In case of MC-dropout, the only difference is that multiple outputs are provided for a single input wherein uncertainty estimation is desirable. Furthermore, this method provides higher independency between outcomes that reflects the behaviour of the variety of decision functions.

2.2 Predictive Uncertainty Using MC-Dropout

Several uncertainty measurements have been presented in the previous studies such as variance, mutual information and others. We only consider mutual information (MI) to express predictive uncertainty. Mutual information distinguishes the type of uncertainty by capturing the epistemic uncertainty in the model [8]. Referring to [9], mutual information (MI) can be expressed as:

$$\hat{I}(y^*|x^*, w) = \hat{H}(y^*|x^*, w) + \sum_c \frac{1}{T}\sum_{t=1}^{T} p(y^* = c|x^*, w)\log p(y^* = c|x^*, w) \tag{6}$$

where c is the class label, and

$$\hat{H}(y^*|x^*, w) = -\sum_c p_{MC}(y^* = c|x^*, w)\log p_{MC}(y^* = c|x^*, w) \tag{7}$$

MI captures the mutual dependency between the hypotheses derived from Monte-Carlo samples over the predictions in which MI identifies the information gain of the model's confidence about its output.

2.3 Uncertainty and Risk

An opposed discussion regarding MC-dropout has been introduced in [11]. Risk is defined as the inherent variability in a model, whereas uncertainty appears to capture our uncertain belief about the predicted value. Consider the popular example of tossing a coin. The probability of drawing head or tail will hold some risk. However, in the case of a biased coin, our beliefs are updated when we get more observations which are denoted as the uncertainty. This "risk" term matches the case when dealing with feature vectors falling near the decision boundary in binary classification tasks. Consequently, MC-dropout provides flipped outputs across the extremely close classes. Furthermore, the test points occurring between two overlapping classes also identify aleatoric uncertainty. Thus, the two terms risk and aleatoric uncertainty hold the same concept to some extent. In both types, the risk in the model and aleatoric uncertainty of the predictions are irreducible. This example will be illustrated later in the experiment of this paper.

3 Experiments and Discussions

3.1 Classification Using Synthetic Data: Toy Example

In this section, we provide an empirical study of MC-dropout method using synthetic datasets for classification task. We perform two experiments (separable and non-separable data) using two different datasets generated by "make_classification" and "make_circles" functions provided by scikit-learn package in Python Programming Language [18]. "make_classification" is used to randomly generate a balanced n-class distributions drawn from a normal distribution situated on the vertices of 2-D hypercube with length 2 (using two clusters per class). "make_circles" produces Gaussian samples with a spherical decision boundary in which the function's parameters acquire noise of 0.1 and scale factor of 0.5 to separate the circles. For separable and non-separable data, we generate 1000 samples of 2-D features corresponding to two different classes altogether as shown in Fig. 1. We train a neural network for each dataset with same settings of two hidden layers of widths 100 and 81 respectively, and an output layer squashed with a softmax function for the binary classification. A dropout function is applied after every hidden layer with dropout ratio of 0.4. After that, the same set is used in testing with activated dropout setting to obtain the Monte-Carlo samples with 100 iterations (T = 100). The learning rate is set to 0.001 and the number of epochs is fixed at 50. The average of the Monte-Carlo samples is used as the predicted mean referring to Eq. 5, and its predictive uncertainty is computed using MI. In the upcoming experiments, we denote by uncertainty threshold as T_u as introduced previously in [9]. This threshold can be randomly assigned between the minimum and the maximum uncertainty values in the test set.

Using data of "make_classification". Using the data of "make_classification", we compute the predictive mean and uncertainty of each instance as shown in Fig. 2. Clearly, the data points with predictive mean in a neighbourhood of 0.5 are more likely to acquire higher predictive uncertainty. We assign different thresholds $\boldsymbol{T_u}$ to the predictive MI,

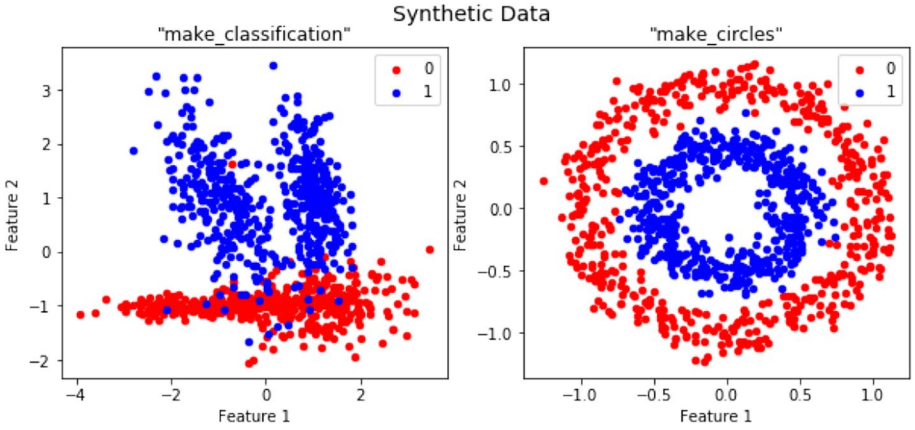

Fig. 1 Synthetic data. Toy example of separable (left subplot) and non-separable (right subplot) datasets

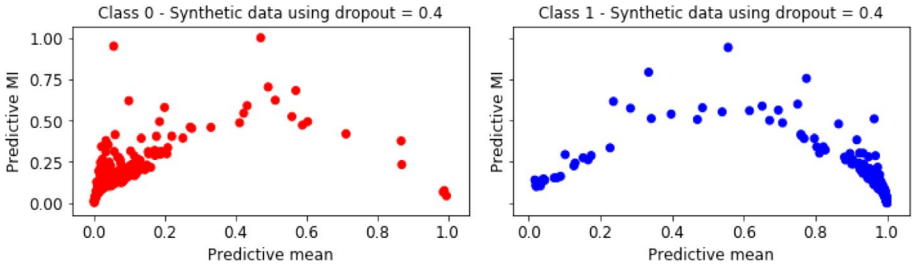

Fig. 2 Data derived from "make_classification". Predictive mean versus MI of class 0 (red) and class 1 (blue)

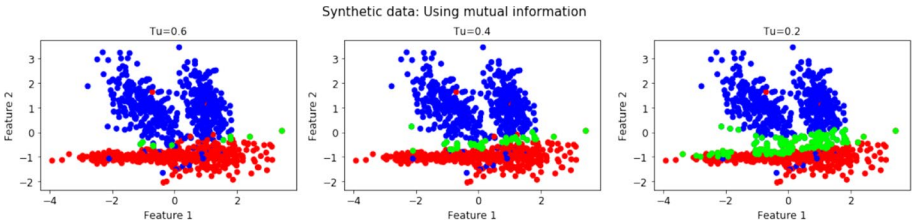

Fig. 3 Synthetic data with different uncertainty thresholds $T_u = \{0.6, 0.4, 0.2\}$ respectively in the subplots. Class 0 is plotted in red and class 1 is in blue. The green plots belong to the uncertain instances with respect to T_u using predictive MI

where $\boldsymbol{T_u} \in \{0.6, 0.4, 0.2\}$. The data points with MI values exceeding $\boldsymbol{T_u}$ are highlighted in which uncertain predictions occur. Subsequently, we highlight the plottings of uncertain predictions assigned by these thresholds as depicted in Fig. 3. Interestingly, these uncertain instances are located between the boundaries of the class distributions. Starting with $\boldsymbol{T_u} = 0.6$, a few green points that correspond to uncertain predictions lies between class 0 (red) and class 1 (blue) as shown in Fig. 3.

A higher number of green points follows the boundary patterns as more uncertain points are added with the increased threshold. Specifically, these green instances show different types of uncertainty. Referring to Fig. 3, the highlighted instances falling between the transition region of the opposed classes are known as of aleatoric uncertainty type. The other green points that are not falling in the transition region of the class distributions are rather falling on the edge of the class itself in which the model has a weak belief about the predictions (shown in the most left and right sides of each subplot in Fig. 3). These points correspond to the model's epistemic uncertainty where the lack of training points occur as introduced in the related work. Apart from the major types of uncertainties, we realise that MC-dropout captures the points falling on the boundaries of the class distributions. Thus, MC-dropout allows the variability in the model's parameters (ensemble of hypotheses) resulting in variations of the boundary lines between class distributions. For brevity, we generate three different test points of coordinates (−0.5,4), (−3,0), (0, −0.5) as shown in Fig. 4.

Afterwards, we perform multiple stochastic passes to compute MI of each case, in which MC-dropout behaviour can solely be summarised in three different cases as following:

1. First test point of coordinates (−0.5, 4): The predictive MI of this point is 0. This case reveals the drawback of MC-dropout in which the test point lacks to training data and cannot be detected as epistemic uncertainty as it is far from the decision boundary between the given classes.
2. Second test point of coordinates (−3, 0): This point has acquired predictive MI of value 1. This point falls near the decision boundary of the given classes, which causes variability in the decision functions using monte-carlo samplings.
3. In third case, the point of coordinates (0, −0.5) is generated on region of overlapping classes (decision boundary). The predictive MI at this point is 0.229. The reason of low uncertainty on the decision boundary is due the limited variability of the line separating these two classes, whereas the test point occupies a region of overlapped classes. Hence, the multiple passes on this point provide nearly stable predictions.

Using data of "make_circles". In this experiment, we apply MC-dropout on non-linearly separable data generated by "make_circles" function. Similarly, we choose different values of T_u of 0.6, 0.4 and 0.2 to study the effect of MC-dropout using predictive MI as shown in Fig. 5.

As depicted in Fig. 6, MI apparently captures the data falling on the spherical boundary between the classes distributions. Furthermore, it is reasonable to obtain more uncertain instances that belong to the class 0 (outer circle in red). This issue is due to the dispersed

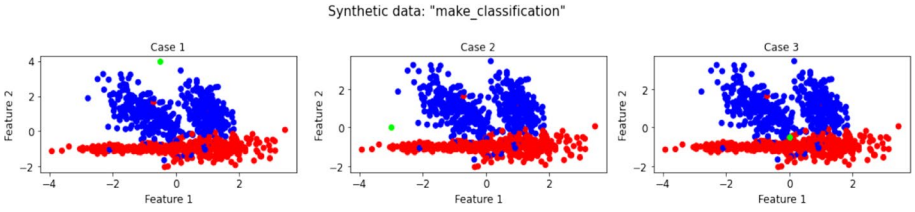

Fig. 4 Representation of "make_classification" data. Each subplot represents the given training set with a single test point shown in green that reflects a unique behaviour of MC-dropout

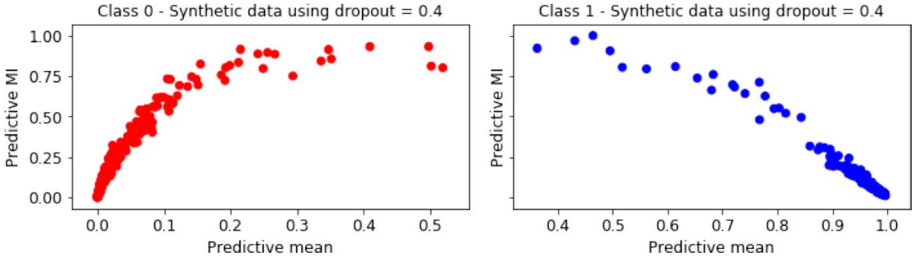

Fig. 5 Data derived from "make_circles". Predictive mean versus MI of class 0 (red) and class 1 (blue)

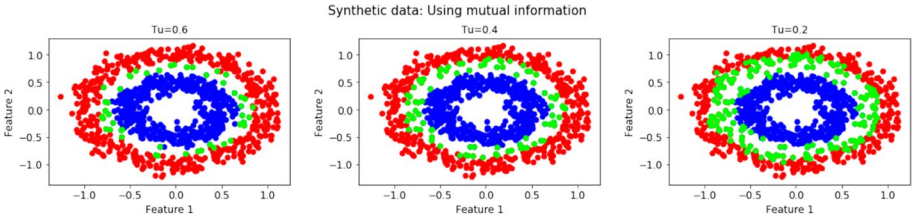

Fig. 6 Synthetic data with different uncertainty thresholds $T_u = \{0.6, 0.4, 0.2\}$ respectively in the subplots. Class 0 is plotted in red and class 1 is in blue. The green plots belong to the uncertain instances with respect to T_u using predictive MI

distribution of class 0 on a wider circle which increases the variability of decision boundary resulting in weak predictions.

In general, there is no doubt that part of the uncertain instances captured by MC-dropout belongs to aleatoric uncertainty type and others for epistemic type. However, the original behaviour of this method is not tied with uncertainty rather than the variability of the decision boundary. For instance, the outer circumference of the outer circle in Fig. 6 does not include any type of uncertainty. Therefore, any data point falling on the outer region apart from any data is provided with high certainty that belongs to class 0 (red) which contradicts the concept of uncertainty estimation.

3.2 Classification Using Real Data: Elliptic Data

In this section, we study the effect of MC-dropout method on real data known as Elliptic data, which is derived from Bitcoin blockchain. In this experiment, we discuss the relative merits of using MC-dropout.

Elliptic data. It is one of the largest publicly available dataset which is derived from Bitcoin blockchain [19]. Specifically, it is a graph of bitcoin transactions with more than 200 k nodes and 334 k edges. The nodes correspond to the transactions between addresses in Bitcoin blockchain, and the edges are the link issued from the source to the destination of these transactions. The Bitcoin transaction graph of Elliptic data acquires partially labelled nodes to distinguish between licit and illicit transactions. The licit transactions belong to the normal behaviour of the addresses such as mining bitcoin and other licit services. The illicit transactions are derived from illicit services of transactions such as money laundering, ransomware, and others. The partial labelling of the graph nodes is acquired

Table 1 Comparison between standard and MC-dropout models using Elliptic data

Model	Accuracy	F_1-score	AUC
Standard (no dropout)	0.943	0.619	0.892
MC-dropout	0.966	0.729	0.894

Area under curve (AUC) represents the goodness of classification between class 0 (licit) and class 1 (illicit)

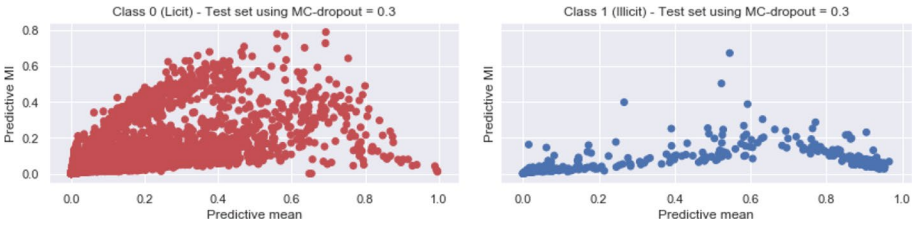

Fig. 7 Predictive mean and MI of the test set using Elliptic data. The left subplot corresponds to licit class (red) whereas the right subplot corresponds to the illicit class (blue)

using heuristics based reasoning process [19]. This data is accompanied by 166 features, in which the first 94 corresponds to local features including the timestamp, and the rest corresponds to the aggregated features acquired by the specificity of the graph network. Mainly, this data is suitable for binary classification in machine learning to assist anti-money laundering processes by spotting the licit and illicit nodes. Elliptic data is a collection of distinct transaction graphs with 49 timestamps, where each of these graphs falls in a separated timestamp. Moreover, the number of nodes per timestamp is approximately uniform in each graph. This data is highly imbalanced with 21% of nodes are labelled licit, and only 2% are illicit, where the other labels are of unknown identity. In what follows, we only consider local features excluding timestamps which counts to 93 features.

MC-Dropout with Elliptic Data. Using Elliptic data, the first 34 graphs are used as a training set including a validation set, whereas the remaining 15 graphs are used as a test set. We assess the performance of MC-dropout by training two neural networks with and without dropout each of two hidden layers of 100 and 81 neurons respectively. We refer to the first model as standard model, where dropout layer is not involved in the algorithm. The second model, MC-dropout model, is tied with dropout layer after every hidden layer with dropout ratio of 0.3, wherein monte-carlo sampling is performed on the test set with 100 stochastic forward passes. Hence, predictive uncertainties are measured using mutual information (MI). We evaluate the performance of these two models to reflect the goodness of classification as shown in Table 1. Unsurprisingly, MC-dropout has outperformed a standard model since the former model uses ensemble of decision functions with shared parameters which enhances the final predictions.

Illustration of MC-Dropout Behaviour on Elliptic Data. Regarding MC-dropout model, we plot predictive mean versus MI for licit and illicit class as shown in Fig. 7. Referring to Fig. 7, the two subplots refer to the true labels of class licit and illicit respectively, whereas the predictive mean corresponds to the model's predictions after using multiple stochastic forward passes. Generally, we realise that false predictions are associated with high certainty as the model is able to detect few with uncertain estimations. There are two assumptions behind the high certainty of false predictions. The first assumption is

that these points are similar to case 1 in the previously mentioned experiment of synthetic data. The second assumption could be the lack of features, wherein the test points are erroneously embedded in the wrong class, and these predictions cannot be detected by any machine learning model. On the other hand, assuming that a threshold T_u of 0.5 is assigned to the given test set, and we reject the uncertain test points. Consequently, the accuracy of the accepted predictions (remaining test points) becomes 0.967 with F_1-score 0.736.

Out-of-Distribution Data. To represent Out-of-Distribution (OoD) data, we generate 100 test points with 93 features each sampled from Gaussian distribution of mean 3.0 and standard deviation 1.0 chosen arbitrarily, knowing that the original data is normalised. With no doubt, MC-dropout is not able to provide any uncertainty on these points, in which they are predicted as licit class with predictive uncertainty equals to zero.

4 Conclusion and Future Work

Dealing with uncertainty estimates is a desirable approach when exposed to critical predictions such as the case in anti-money laundering, where licit and illicit services are involved. MC-dropout method as Bayesian approximation is able to capture a portion of epistemic and aleatoric uncertainty types, however this method is rather capturing data points falling on the decision boundary of the given classes. This method is not a reliable approach for out-of-distribution data and does not differ from unjustified predictions provided by a standard neural networks. Nonetheless, MC-dropout deals efficiently with misclassified instances located between the boundaries of different given classes, which is highly viewed in image datasets. In the future work, we foresee uncertainty predictions via distances from a given test point to a randomly chosen nearest neighbours, in which epistemic uncertainty can be identified.

Acknowledgements This work is supported by Bournemouth University. Data is publicly available under a Public License thanks to Elliptic company (www.elliptic.co).

Funding Funding was provided by Bournemouth University.

References

1. Shen D, Wu G, Suk H-I (2017) Deep learning in medical image analysis. Annu Rev Biomed Eng 19:221–248
2. Rawat W, Wang Z (2017) Deep convolutional neural networks for image classification: a comprehensive review. Neural Comput 29(9):2352–2449
3. Khan S, Yairi T (2018) A review on the application of deep learning in system health management. Mech Syst Signal Process 107:241–265

4. Najafabadi MM, Villanustre F, Khoshgoftaar TM, Seliya N, Wald R, Muharemagic E (2015) Deep learning applications and challenges in big data analytics. J Big Data 2(1):1
5. Neal RM (2012) Bayesian learning for neural networks, vol 118. Springer, Berlin
6. Gal Y, Ghahramani Z (2016) Dropout as a bayesian approximation: representing model uncertainty in deep learning. In: international conference on machine learning, pp 1050–1059
7. Kendall A, Gal Y (2017) What uncertainties do we need in bayesian deep learning for computer vision? In: Advances in neural information processing systems, pp 5574–5584
8. Smith L, Gal Y (2018) Understanding measures of uncertainty for adversarial example detection, arXiv preprint arXiv:1803.08533
9. Mobiny A, Nguyen HV, Moulik S, Garg N, Wu CC (2018) Dropconnect is effective in modeling uncertainty of bayesian deep networks, arXiv preprint arXiv:1906.04569
10. Gal Y (2016) Uncertainty in deep learning, Ph.D. thesis, University of Cambridge
11. Osband I (2016) Risk versus uncertainty in deep learning: Bayes, bootstrap and the dangers of dropout. In: NIPS workshop on bayesian deep learning, vol 192
12. Buntine WL, Weigend AS (1991) Bayesian back-propagation. Compl Syst 5(6):603–643
13. MacKay DJ (1992) A practical bayesian framework for backpropagation net-works. Neural Comput 4(3):448–472
14. Graves A (2011) Practical variational inference for neural networks. In: Advances in neural information processing systems, pp 2348–2356
15. Neal RM (1992) Bayesian training of backpropagation networks by the hybrid monte carlo method, Tech. rep., Citeseer
16. Neal RM (1993) Bayesian learning via stochastic dynamics. In: Advances in neural information processing systems, pp 475–482
17. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
18. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Pas-sos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830
19. Weber M, Domeniconi G, Chen J, Weidele DKI, Bellei C, Robin-son T, Leiserson CE (2019) Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics, arXiv preprint arXiv:1908.02591